CLAIMS

What is claimed is:

1. A method for code completion, comprising:

providing a representation of a first program in a first programming language;

establishing a location in the first program;

associating the location with a representation of the first program;

obtaining code completion information relevant to the location in the first program based on the representation of the first program; and

wherein the obtaining occurs at the behest of an extensible compiler framework.

2. The method of claim 1 wherein:

the location in the first program is one of: 1) a textual offset; 2) a structural navigation through a parse tree; 3) at least one semantic entity in the first program; and 4) a token or token range.

3. The method of claim 1 wherein:

the representation of the first program is a parse tree.

4. The method of claim 3 wherein:

the code completion information is based on information related to a node in the parse tree.

5. The method of claim 1 wherein:

the code completion information includes at least one of: 1) a class name and/or definition; 2) a type name and/or definition; 3) a field/member/variable name and/or definition; 4) a method name and/or definition; and 5) a function name and/or definition.

6. The method of claim 1, further comprising:

analyzing the syntactic structure of a first program in a first programming language, wherein the first program can be represented by a first set of tokens;

7. The method of claim 1 wherein:

the extensible compiler framework can integrate and interact with compilers for different programming languages through a common interface.

8. The method of claim 1 wherein:

the first program in the first programming language can be nested within a second program in a second programming language.

9. The method of claim 1 wherein:

a second program in a second programming language is nested within the first program in the first programming language.

10. A system comprising:

means for providing a representation of a first program in a first programming language;

means for establishing a location in the first program;

means for associating the location with a representation of the first program;

means for obtaining code completion information relevant to the location in the first program based on the representation of the first program; and

wherein the obtaining occurs at the behest of an extensible compiler framework.

11. A system for code completion, comprising:

a component operable to provide a representation of a first program in a first programming language;

a component operable to establish a location in the first program;

a component operable to associate the location with a representation of the first program;

a component operable to obtain code completion information relevant to the location in the first program based on the representation of the first program; and

wherein the obtaining occurs at the behest of an extensible compiler framework.

12. The system of claim 11 wherein:

the location in the first program is one of: 1) a textual offset; 2) a structural navigation through a parse tree; 3) at least one semantic entity in the first program; and 4) a token or token range.

13. The system of claim 11 wherein:

the representation of the first program is a parse tree.

14. The system of claim 13 wherein:

the code completion information is based on information related to a node in the parse tree.

15. The system of claim 11 wherein:

the code completion information includes at least one of: 1) a class name and/or definition; 2) a type name and/or definition; 3) a field/member/variable name and/or definition; 4) a method name and/or definition; and 5) a function name and/or definition.

16. The system of claim 11, further comprising:

a component operable to analyze the syntactic structure of a first program in a first programming language, wherein the first program can be represented by a first set of tokens;

17. The system of claim 11 wherein:

the extensible compiler framework can integrate and interact with compilers for different programming languages through a common interface.

18. The system of claim 11 wherein:

the first program in the first programming language can be nested within a second program in a second programming language.

19. The system of claim 11 wherein:

a second program in a second programming language is nested within the first program in the first programming language.

20. A machine readable medium having instructions stored thereon that when executed by a processor cause a system to:

provide a representation of a first program in a first programming language;

establish a location in the first program;

associate the location with a representation of the first program;

obtain code completion information relevant to the location in the first program based on the representation of the first program; and

wherein the obtaining occurs at the behest of an extensible compiler framework.

21. The machine readable medium of claim 20 wherein:

the location in the first program is one of: 1) a textual offset; 2) a structural navigation through a parse tree; 3) at least one semantic entity in the first program; and 4) a token or token range.

22. The machine readable medium of claim 20 wherein:

the representation of the first program is a parse tree.

23. The machine readable medium of claim 22 wherein:

the code completion information is based on information related to a node in the parse tree.

24. The machine readable medium of claim 20 wherein:

the code completion information includes at least one of: 1) a class name and/or definition; 2) a type name and/or definition; 3) a field/member/variable name and/or definition; 4) a method name and/or definition; and 5) a function name and/or definition.

25. The machine readable medium of claim 20, further comprising instructions that

when executed cause the system to:

analyze the syntactic structure of a first program in a first programming language, wherein the first program can be represented by a first set of tokens;

26. The machine readable medium of claim 20 wherein:

the extensible compiler framework can integrate and interact with compilers for different programming languages through a common interface.

27. The machine readable medium of claim 20 wherein:

the first program in the first programming language can be nested within a second program in a second programming language;

28. The machine readable medium of claim 20 wherein:

a second program in a second programming language is nested within the first program in the first programming language.

29. A method for code completion, comprising:

providing a representation of a first program in a first programming language;

establishing a location in the first program;

associating the location with a representation of the first program;

obtaining code completion information relevant to the location in the first program based on the representation of the first program;

wherein the obtaining occurs at the behest of an extensible compiler framework; and

wherein the extensible compiler framework can integrate and interact with compilers for different programming languages through a common interface.

30. The method of claim 29 wherein:

the location in the first program is one of: 1) a textual offset; 2) a structural navigation through a parse tree; 3) at least one semantic entity in the first program; and 4) a token or token range.

31. The method of claim 29 wherein:

the representation of the first program is a parse tree.

32. The method of claim 31 wherein:

the code completion information is based on information related to a node in the parse tree.

33. The method of claim 29 wherein:

the code completion information includes at least one of: 1) a class name and/or definition; 2) a type name and/or definition; 3) a field/member/variable name and/or definition; 4) a method name and/or definition; and 5) a function name and/or definition.

34. The method of claim 29, further comprising:

analyzing the syntactic structure of a first program in a first programming language, wherein the first program can be represented by a first set of tokens;

35. The method of claim 29 wherein:

the first program in the first programming language can be nested within a second program in a second programming language.

36. The method of claim 29 wherein:

a second program in a second programming language is nested within the first program in the first programming language.